A reference model-driven Architecture linking Business Processes and Services

Andrea Delgado
Instituto de Computación, Facultad de Ingeniería,
Universidad de la República, Uruguay
adelgado@fing.edu.uy

Francisco Ruiz, Ignacio García-Rodríguez de Guzmán
Instituto de Tecnologías y Sistemas de Información,
Universidad de Castilla-La Mancha, Spain
{francisco.ruizg, ignacio.grodriguez}@uclm.es

Abstract

While an important area of research in the services community is focused on the composition of services to orchestrate processes, based mainly on the WS-BPEL standard with a services view, we take a different standpoint. Ours has also a business perspective, defining services to support the execution of business processes (BPs) in a BPMS platform. Our vision focuses on process and service models and the relationships between them, allowing traceability between elements belonging to the business and the software areas. This paper presents a reference model-driven architecture linking BPs to services, automating the generation of Service-Oriented Architectures (SOA) from BPs architectures. Our approach applies both to collaborative and orchestration BPs; in the first scenario it helps support the agreements reached regarding interactions between different organizations. We believe the application of our model-driven approach, which is based on metamodels, models and transformations between them, can support organizations in their effort to align their BPs with service support.

1. Introduction

The complexity of current organizations and the software systems supporting their day-by-day operation has highlighted the need to develop more flexible systems, both in the academic area and in the field of industry. These systems should allow several views and different technologies to be integrated and to act together as a whole, to overcome the difficulties that most system integration presents. One reason for this lies in the fact that traditional systems were developed mainly with a vertical vision that was based on organizational units or areas in the organization, making the integration of such systems require a great amount of effort from the Information Technology (IT) area. However, the expectations of the business area were rarely fulfilled, either functionally, or in terms of budget and costs, producing what is known as the business-system gap [1][2].

On the other hand, a horizontal vision of the organization is promoted by the Business Process Management (BPM) paradigm [3][4][5]. It is based on the definition and modeling of the business processes (BPs) that the organization carries out, crossing several organizational units. BPM has been used by the business area to define, manage, optimize and improve its BPs, based on the BPs lifecycle [4]. It has been embraced also by the software area, as BPM has gained more importance for organizations that want to explicitly control their way of doing business by means of their BPs and the software to support them.

A key challenge organizations face nowadays involves their ability to react quickly to changes either to their BPs or to the software supporting them [1][2]. These changes can come from different sources: i) external requirements from partners or the market; ii) internal new requirements for the way that things are being carried out; iii) improvement opportunities that were detected from evaluating BPs execution. In addition, the possibilities provided by Internet and globalization have raised several challenges to the way organizations usually conduct their business, as well as to the manner in which they can interact with others.

The Service Oriented Computing (SOC) [6] vision promotes the development of systems based on software-service elements with high cohesion and low coupling, well defined interfaces, and contracts specifying their functionality [1][2]. An intermediate layer of services between BP models and their implementation helps provide a better response to changes, allowing a more agile way to introduce changes, both in BP models and in their implementation, with minimum impact between each other [1][2]. Modeling of both BPs and services is a key aspect in supporting the horizontal vision of the organization and in explicitly their elements, helping provide traceability between one area to the other, so easing the analysis of the impact of changes [1][2].

The Model Driven Development (MDD) [7][8][9] approach, which uses models, metamodels and the

URI: http://hdl.handle.net/10125/50475
ISBN: 978-0-9981331-1-9

transformations between them as a basis for the software development process, can help provide a focus on BPs and service models. With a focus on modeling, business people can do what they know best: they can specify the BPs in the organization. Software people can derive automatically from these BP models the service models to support their execution. In the context of collaborative BPs between organizations, the ability to integrate changes in an independent way, both in the BP models and the technology implementing them, is even more important than it is to a single organization.

In this paper, we present a model-driven architectural approach that focuses on defining a Service Oriented Architecture (SOA) to support BPs architecture, to guide organizations in their efforts to model and execute their BPs and services in a systematic way. In previous work [10][11] we have presented the details of the automated generation of SoaML service models from BPMN 2.0 models, for collaborative [10] and orchestration [11] BPs. We have also added QoS characteristics to service models [12] to enhance the code generation.

The main contribution of this paper is the specification of the overall architectural approach and levels, presenting the reference model-driven architecture that provides the basis for the SOA generation from BPs architecture. We also revise our tool support for modeling and code generation, as well as illustrating the use of the reference architecture and automated generation of service models from BP models. We provide an example for cases both of collaborative and of orchestration BPs.

The rest of the document is organized as follows: in Section 2 we present the motivation for our research, including the research question, challenges and objectives. Section 3 provides a description of our model-driven service-oriented approach for realizing BPs based on a reference model-driven architecture that relates BPs to services. In Section 4 we present an example of the application of our proposal. Section 5 is where we discuss related work and finally, in Section 6, we draw some conclusions and discuss future work.

2. Motivation and research objectives

Models have been proven to play an important role in the software development process: informally, they sketch out the concepts of the system for evaluation of solutions and communication among stakeholders; in a more formal way, they specify them without ambiguity, with metamodels, models and transformations between models, as in MDD. When models are used informally, only to sketch out the concepts of the system under development, they are not the basis of development; they provide support mainly for communication among stakeholders, as well as for clarification of ideas and evaluation of existing solution options.

When models become the center of development, the need to specify them correctly and without ambiguity increases, as syntactic incompleteness makes it impossible to process models [13]. The explicit modeling of BPs in organizations enables them to think about their way of doing business, while also helping them discover weaknesses in their processes. One of their key uses in the context of BP realization by services is that of designing services at a more abstract level than with specific technologies, also promoting traceability between elements [13]. The explicit tracing of the relationships between elements in different models promotes the reuse of the knowledge imbibed in the transformations, in our case between the business and software architectures.

The Business Process Model and Notation 2.0 (BPMN2) [14] standard provides a rich palette of elements to specify BP models, as does the Service Oriented Architecture Modeling Language (SoaML) [15] for adding specific service-related stereotypes to UML models for the functional modeling of services. The Quality of Services (QoS) & Fault Tolerance (QFTP) [16] standard allows specific QoS stereotypes to UML models to be added, for the modeling of non-functional requirements. In the context of collaborative BPs between organizations, defining the notations to use and the interaction points that each BP has to provide in order to interchange information with other participants are key tasks that must be done accurately. These interaction points have to be preserved as much as possible when changes arise, to minimize their impact on the interaction. But we cannot control what changes are needed or where changes are needed, so we can only try to prevent their impact.

This is why an automated way to generate service models from BP models can help. First of all, because the automatic generation of services directly from BPs is based on the definition of correspondences between elements in each metamodel, and so registers knowledge for service design from BPs. This reduces design errors which occur if a designer must think each time about how to define services to support BPs. It also improves productivity of software developing teams, for the same reason. When service models can be generated for the BP collaboration on which the participating organizations have agreed, each one can use the same model as input to obtain their service models. That makes the software development process easier, avoiding problems such as labeling discrepancies in the interaction points (i.e. a name in the provided service and a different one in the

consumer). In addition, it promotes technology-independence and interoperability, since service code can be obtained from a unique generated SoaML service model to different platforms, as required by each organization (WS, JEE, .NET). Based on these and on a literature review [17] which we carried out regarding existing work about automating the implementation of BPs with services (c.f. Section 6), we have defined the following research question and objectives to guide our work:

How can a reference model-driven architecture for automating the generation of service-oriented models from BP models be defined to facilitate the design of services to implement BPs and to support the effective separation of BPs modeling and implementation?

Given the breadth of the question, we broke the original question down into three separate questions:

i) what elements are needed to define a reference model-driven architecture for automating the generation of service oriented models from BP models? (i.e., research question RQ1)

ii) what tools should be available to support the definitions in RQ1 and facilitate the design of services to implement BPs? (i.e., research question RQ2)

iii) are the proposals made in RQ1 and RQ2 appropriate and useful in easing the design of services to implement BPs and to support the effective separation of BPs and service modeling and implementation? (i.e., research question RQ3)

To be able to provide detailed answers to the research questions, we stated the following specific objectives, which are related to them: to answer RQ1 (i) analyze and define concepts and relationships between BPs and service models; (ii) define mappings for concepts between the BPMN 2.0 and SoaML metamodels, along with QVT transformations for implementing them; to answer RQ2 (iii) provide tool support for the definitions in (ii) and code generation facilities; and to answer RQ3 (iv) provide evidence (empirical validation) of the applicability of the approach (not presented here).

It is worth noting that in (i) and (ii) we aim to provide a (potentially partial) answer to a very difficult and traditionally open issue, namely that of the relationship between the requirements defined for a system and the corresponding software design: i.e. to relate the problem space (business area in the form of BPs architecture) to the solution space (software area, in the form of service oriented architecture, SOA). To answer (i) we have defined an ontology to identify and relate concepts that can be seen in [18], and in (ii) we applied it to BPMN 2.0 and SoaML metamodel elements, to define mappings between corresponding elements. To the best of our knowledge, our approach is the only one which explicitly relates BP and service models and which is completely based on standards and models.

These definitions, mappings and transformations constitute the answers to RQ1, and are set out in Sect. 3.2. To answer RQ2, we have developed new tools and integrated existing ones for supporting the modeling of BPs and services and associated code (i.e. SoaML Toolkit), as well as tools for defining and executing transformations (i.e. MediniQVT/Eclipse ATL) and this is set out in Sect. 3.3

To address the research work, we applied a combination of research methods suited to each research phase. Firstly, we carried out a systematic literature review to identify existing proposals, as mentioned (cf. [17]). Elements presented as part of the RQ1 answer were identified by means of an extensive analysis of existing standards and approaches for modeling BPs and services (cf. [10]). For the development of our proposal, we followed design science principles as suggested by [19], creating a set of artifacts: a) mappings and transformations between BPMN 2.0 and SoaML elements [10][11] to automate the generation of service models from BP models as part of the RQ1 answer b) a tool chain to support the approach (i.e. SoaML Toolkit as our own development, and an existing QVT engine) as the RQ2 answer.

3. Reference model-driven architecture

Our approach follows the Model Driven Architecture (MDA) [20] principles, and is based completely on the use of the OMG standards: BPMN 2.0 [14], SoaML [15], QoS [16] and Query/View and Transformations (QVT) [21]. Fig. 1 presents our reference model-driven architecture on an MDA view, relating concepts from the BP Architecture based on BP models on the left side, to the Service Oriented Architecture (SOA) based on service models on the right side.

MDA defines three types of models: i) Computation Independent Model (CIM), corresponding to models specifying the requirements for the system, e.g. UML models such as Use Case models or BPMN2 BP models; ii) Platform Independent Model (PIM) corresponding to design models specifying the solution to the problem, e.g. UML design classes or SoaML service models, and iii) Platform Specific Model (PSM) corresponding to implementation models defining the technology, e.g. JEE stereotypes to SoaML or UML class models.

Fig. 1 presents horizontally the three levels of abstraction defined by MDA (CIM, PIM and PSM), and the code level for the execution of BPs realized by services. Vertically, it presents the two areas we are relating: the Business Process Architecture definition on the left side, and the Service Oriented Architec-

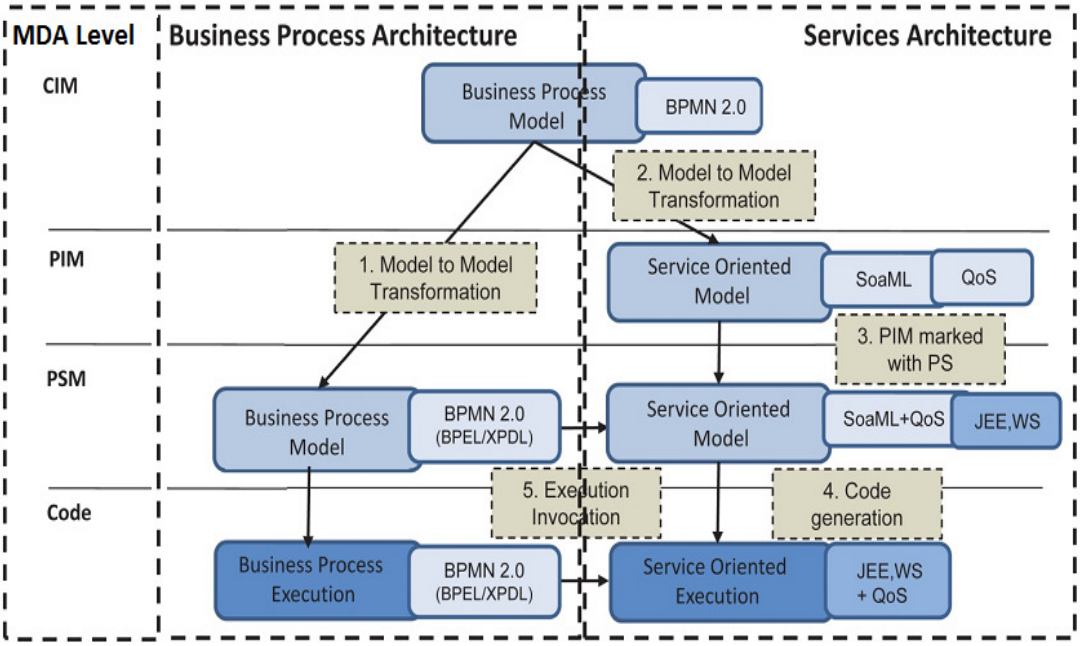

Figure 1 Reference model-driven Architecture from BPs to Services based on MDA

ture (SOA) definition on the right side, stating the level of abstraction in which each model is located, and the transformations defined from one level to another.

As defined in [5] a Process Architecture is a conceptual model showing the BPs of the organization at different levels of abstraction, along with their relationships. At Level 1 – "Process Landscape" a map of the organizational processes is provided. Level 2 – "Abstract Process Models" provides a high-level description of the BPs identified, and finally at Level 3 – "Detailed Process Models" those processes are explicitly modeled, using for example BPMN 2.0.

On the BPs Architecture side of Fig.1, we are standing on Level 3, so the BP models are specified in BPMN 2.0, where the CIM BP model is also the PIM by means of the Identity transformation (not included in the figure). When the notation is BPMN 2.0, the PSM model is also the PIM BP model, but extended with implementation details. If the language for execution is other than BPMN 2.0 (e.g. XPDL/WS-BPEL) a transformation must be performed, for which existing approaches are used, rather than another one being provided. To add invocations to the generated services in the executable BP model, taking the PSM model as input, we insert the invocation into the service activities definition, using the information in the SoaML model. Note that as mentioned above, several other elements have to be added and/or defined in the BPMN2.0 model to make it executable. These may include components such as implementing user forms, scripts, or business rules, for which manual and/or assisted development work is required (depending on the selected execution platform).

Going down the Service Oriented Architecture side, the service models we have defined are shown, where the SoaML service model constitutes the Platform Independent Model (PIM) which is generated automatically from the BPMN 2.0 model. The SoaML model is enhanced with QoS characteristics to add non-functional aspects for the generation of services code. From the PIM SoaML model the Platform Specific Model (PSM) is obtained by adding platform specific information, such as that provided by the JEE and WS definitions, to generate the code both for functional and QoS for services.

On the one hand, therefore, the services are generated from the BP model, and on the other hand, the executable BP model is implemented, including the invocation to the generated services. The chain of transformations presented makes it possible to complete the traceability from the BP to its supporting service implementation, providing the information on which activity (or group of them) from the BP model is designed as a service in the SoaML model, as well as the correspondence in the executable BP model between activities with the implemented services.

3.1. Cases for services realizing BPs

Our vision for realizing BPs with services integrates different cases that can arise when executing BPs in a process engine in an organization. Fig. 2 displays such a perspective.

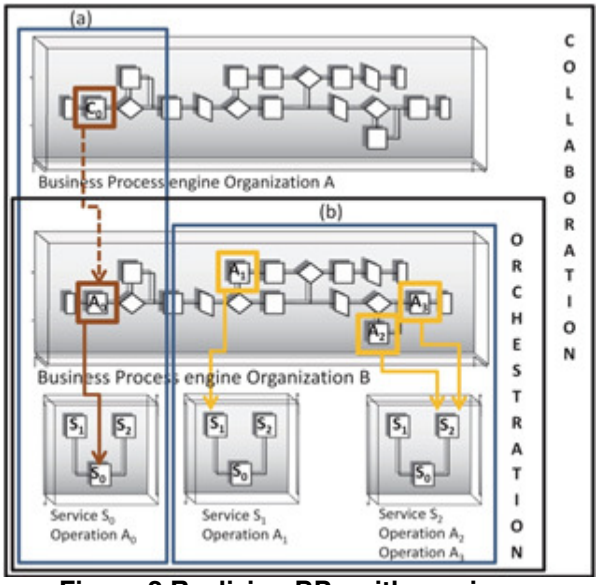

Figure 2 Realizing BPs with services: a) collaborations and b) orchestrations

Fig. 2 shows an organization called A that interacts with an organization called B, by invoking: a) defined services from each other in a collaboration, b) internal services from an organization in an orchestration and c) external services from the cloud (Software as a Service, SaaS) as needed. Rectangles in Fig. 2 are to emphasize which are the invocations marked as a), b) and c). We have used these scenarios to define mappings and transformations, to support each one.

For collaborative BPs (case a) [10] we define service consumers and providers based on the interactions between the participants, and for orchestration BPs (cases b, c) [11] we generate service providers to support the execution of atomic activities (tasks) or grouped activities based on workflow patterns [22]. Fig. 2 shows both cases: (1) in a collaborative BP a consumer C0 from organization A invokes a task A0 from organization B, which in turn invokes the operation A0 from the service S0 generated to support that task; (2) in an orchestration BP we defined two different cases: first, a task A1 is implemented with a service S1 with a single operation A1, as before, and second, a related group of tasks, namely A2 and A3, are implemented with a service S2 with two different operations A2 and A3, generated to support tasks A2 and A3 respectively.

In the case of collaborative BPs, the focus is on generating services to support interactions between participants in different organizations (overall Service Architecture). We have defined a pattern based on MessageFlows of interactions in which a ServiceTask is involved, providing a high level view of the interactions between all participants and services.

In the case of orchestration BPs, the view is restricted to only one participant (internal Service Architecture) -which can be one of the collaboration participants- so the generation is based on the identification of ServiceTasks and on a pattern approach for grouped ServiceTasks in the path of gateways such as AND, XOR and OR, based on [22].

In all cases we generate a SoaML Model from the BP Model, and the ServicesArchitecture, Services and Participants diagrams which correspond to the type of BP we are dealing with.

3.2. Automated generation of SoaML models from BPMN 2.0 models

Fig. 3 shows a collaborative BP model in BPMN 2.0 and the SoaML + QoS services model generated (in different SoaML diagrams), along with the mappings defined between them for the generation:

(1) Pools in BPMN 2.0 to Participants in SoaML with Service ports (both provided and consumed),

(2) Service pattern in BPMN 2.0 (ServiceTask + message flow + invoking Task) to:

- Service Interfaces including operations and parameters (input, output and types)
- ServiceContracts for the identified Services, with information regarding Service Interfaces and roles within the contract (consumer, provider)

(3) and (4) show the QoS added to SoaML Service Ports and Service Contracts respectively, for service specification and agreements.

Rectangles (A) and (B) on the BPMN 2.0 model show the two generation options: (i) interaction pattern between participants for collaborative BPs, and (ii) internal pattern within a single participant for orchestration BPs.

3.2.1. Service generation from Collaborative BPs. Service generation for collaborative BPs [10] covers case a) for invoking services from a partner organization. Our main definition involves identifying a pattern based on a ServiceTask (producer) in one pool, which is invoked by a Task (consumer) in another pool; these are connected by a Message. Based on these, we generate the ServicesArchitecture with Participants from each Process (Pool), an Interface with one Operation and Parameters for the provider from the ServiceTask, together with an Interface for Consumers with one Operation and Parameters, when needed. We also generate the corresponding ServiceContract for each Service provided.

One of the most important contributions of this approach is to provide each organization with the ServiceContracts and Interfaces needed to interact with each other, decreasing the level of misunderstandings.

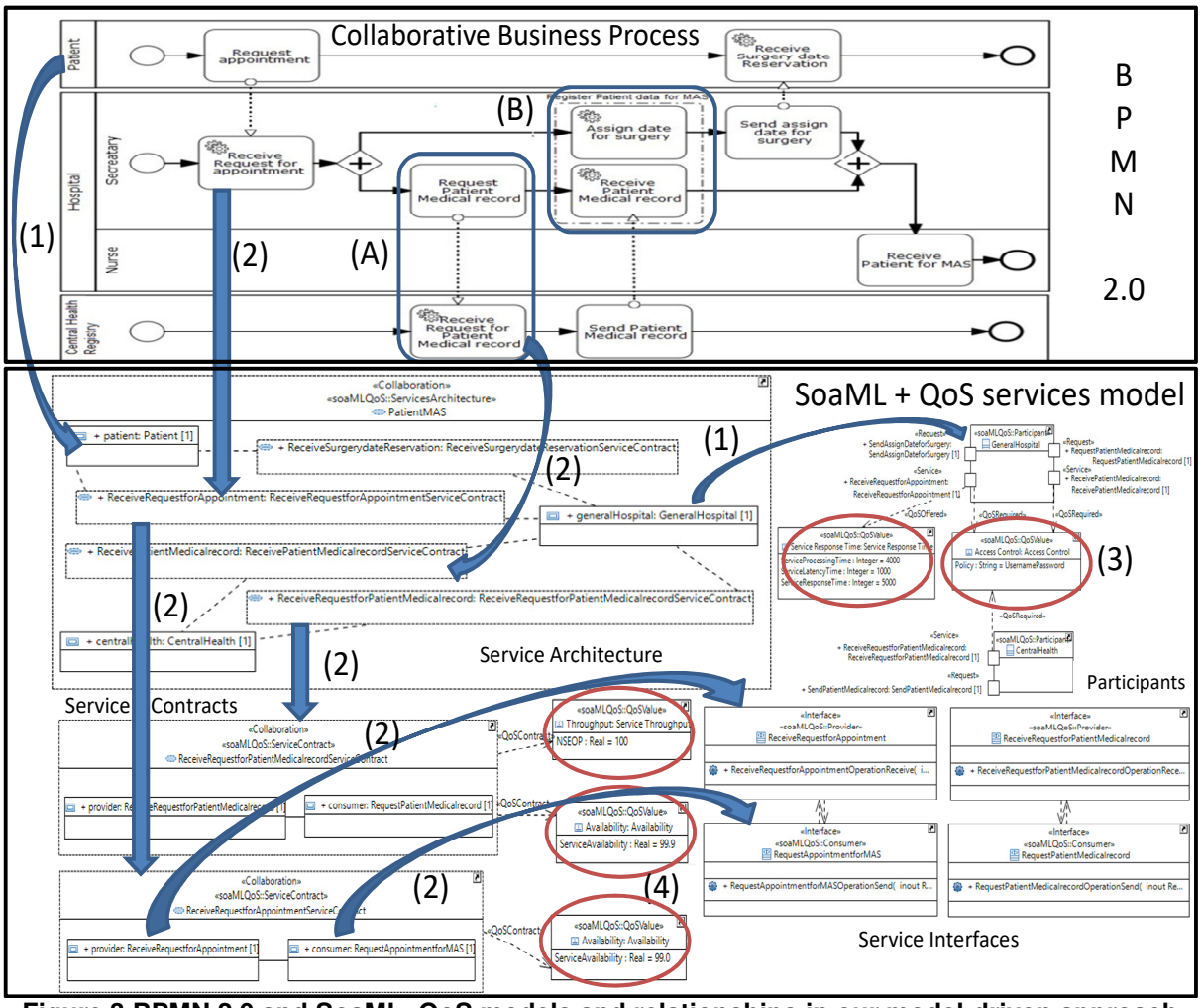

Figure 3 BPMN 2.0 and SoaML+QoS models and relationships in our model-driven approach

3.2.2. Service generation from Orchestration BPs. The generation of services for orchestration BPs [11] covers the rest of the cases: b) internal and c) external service invocation to support the operation of a single organization. In these cases we do not use the pattern presented before, as we are not interested in the collaboration between two organizations, but rather in the implementation of a single process (orchestration) within a single organization (internal Architecture).The Process element thus corresponds to the complete ServiceArchitecture, since we want to generate only the services for the organization modeled by the Pool, as mentioned before; lanes correspond to internal participants which will offer or request services.

In the basic generation approach we obtain a service from a ServiceTask, as before, with one Operation and Parameters. In the patterns approach we use the basic control flow patterns regarding the different types of gateway defined in BPMN 2.0 (AND, XOR, OR). From the ServiceTasks in the paths that belong to a single marked Group, we generate a single Service providing one operation for each ServiceTask, where each of them will be invoked from the corresponding ServiceTask in each gateway path.

3.2.3. QVT transformations. We have defined two types of QVT transformations regarding the SoaML service model output as defined in the standard: (i) unidirectional services and (ii) bidirectional services. For both cases we generate services with simple UML interfaces; for the second we also generate services with the SoaML ServiceInterface. Each transformation is composed of a set of seven rules defined as top relations, and also five other rules that are invoked from these, by defining conditions in their "where" clause that have to be met when executing the rules. Fig. 4 shows the input and output models for the QVT transformation with the top rules, as well as the general structure we have defined for them.

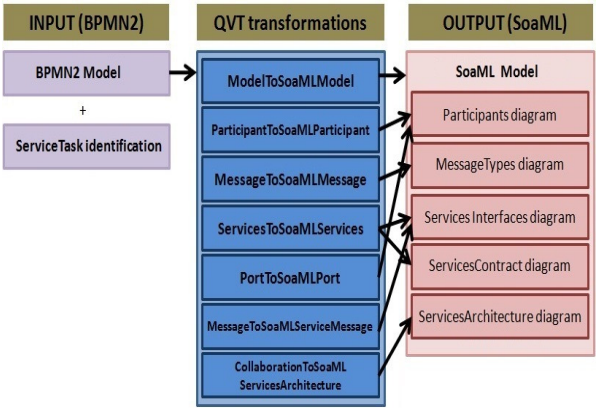

Figure 4 General QVT transformation

3.3. Automated code generation from SoaML + QoS models

From the SoaML service models we can generate the code for each service provided within each participant (Pool, Lane) in both collaborative and orchestration BPs cases. We also provide support for adding QoS to the SoaML model, to complete the specification of services. Details of the complete code generation can be seen in [12].

3.3.1 JEE and WS code generation. We have also defined mappings between elements from the SoaML metamodel and JEE and Web Services definitions, for the generation of the code. A QoS quality model can also be defined to add QoS characteristics to the SoaML services model, to extend the scope of the code generation. Since the code generation is not the focus of the article, we will briefly describe it here.

In the first place, for each Participant in the SoaML service model we generate a Java Participant Project, with the same name as the UML Participant class. From the service Interface a Java Interface is generated, with the corresponding operation/s, as defined in the service. From the SoaML Port a Java class is generated; this implements the service Interface. From Message Types, Java classes which are used as the type of the parameters in the operation/s are also generated. A WSDL document is thus generated that includes all the information of the service.

QoS characteristics can be added to the SoaML service models (e.g. Response Time, Security), both for a service specification and an interaction. In the first case, the QoS is linked to the service Port of the service provider, and in the second case it is linked to the ServiceContract of the interaction and roles. When QoS are added to the SoaML, the code generation takes them into account, adding the corresponding elements to the WSDL and new Java classes to implement the behavior of the characteristic.

3.3.2 Tools support. We have defined our own distribution of the Eclipse environment, and called it Eclipse MINERVA Design[1].It includes a QVT engine for the specification and execution of the transformations–for now MediniQVT[2]– and an integrated tool support we provide in the form of an Eclipse plug-in we called SoaML Toolkit plug-in [12]. This includes: the Eclipse SoaML plug-in for service modeling, and the Eclipse SoaML2Code plug-in to generate the code [23], and two more to support the QoS modeling and the code generation. It does not matter which generation approach we have used (collaborative, orchestration or modeled-from-scratch) to obtain the SoaML model, since it serves as the starting point for code generation. When QoS characteristics are added, we generate the code based on WS* elements.

4. Example of application

As mentioned before, our approach starts from Process Architecture Level 3 [5] with a detailed BP in BPMN 2.0. Our example of application is based on a real business process to perform a "Major Ambulatory Surgery (MAS) for a Patient" from a Hospital in Ciudad Real, Spain; the process is one which we have used from the start of our work. We have simplified it, but have preserved key elements that make the example complex enough and representative enough of the problem we want to solve: i) it is a collaborative process involving several participants (pools), ii) it presents four different occurrences of the Service pattern we transform with our automated generation of services for collaborative BPs (Service Task + message flow + invoking Task) in different pools c.f. section 3.2.1, iii) it presents in the hospital pool the internal control flow pattern for automated generation of complex services (with more than one operation) for orchestration BPs (two service Tasks in different paths within a parallel gateway) c.f. section 3.2.2.

In Fig. 5 the simplified BP model is shown, specified in BPMN 2.0. The MAS Patient requests an appointment for MAS from the Hospital; the Secretary assigns the date and time, and sends the information back to the patient, simultaneously requesting the Patient's medical record from the Central Health Register. On the assigned date the patient arrives at the Hospital and the nurse guides the patient through the rest of the steps, which are not shown.

[1] http://alarcos.esi.uclm.es/minerva/tools/minervadesign.htm
[2] http://projects.ikv.de/qvt

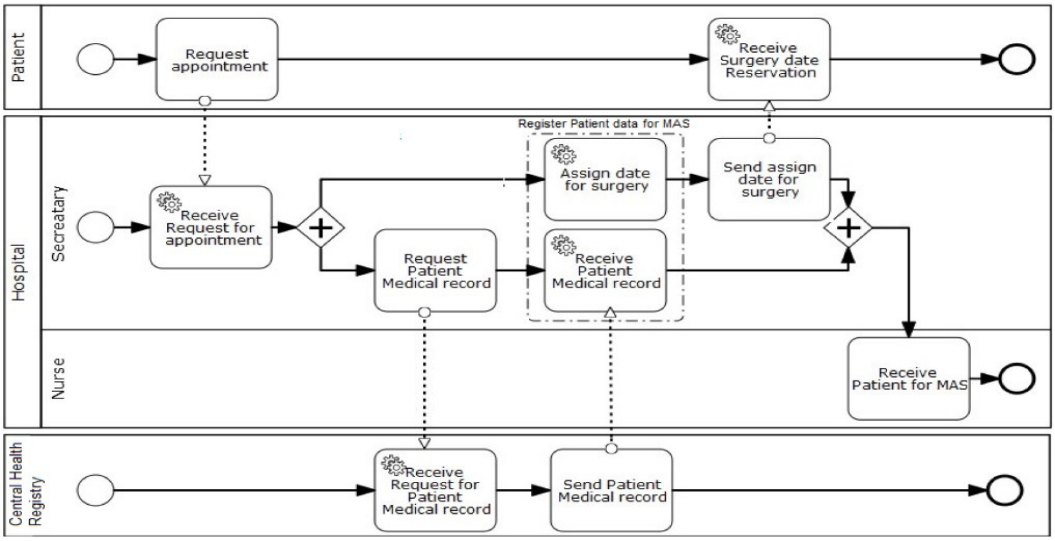

Figure 5 The Patient Major Ambulatory Surgery (MAS) BP from a Hospital

The Collaborative BP has two Pools representing the Participants interacting with the Hospital: one for the Patient, and the other for the Central Health Register. From this BP we first generate the services covering the interaction between participants by applying the QVT transformations for Collaborative BPs. The SoaML service model for this generation includes Participants and services as follows:

- Hospital: "ReceiveRequestforAppointment" and "ReceivePatientMedicalrecord";
- Patient: "ReceiveSurgeryDateReservation"
- CentralHealthRegistry: "ReceiveRequestforPatient Medicalrecord".

Fig. 6 shows some SoaML diagrams, visualized in our Eclipse SoaML plug-in, of the services generated for: (1) collaborative BPs and (2) orchestration BPs.

In the first case in Fig. 6 (1) the diagrams are (a) ServicesArchitecture, showing the three participants: Patient, Hospital and Central Health Register, services, and the roles each participant plays within each service (provider or consumer); (b) an example of a ServiceContract and of a service Interface, and (c) an example of a participant with the associated Ports.

Once the participants have agreed with each other on the "external" services to communicate within the BP execution, each organization has to develop the software support for the internal execution of the BP. Then, for each Orchestration within each Pool, we apply the QVT transformations for Orchestration BPs, in order to generate the internal ServicesArchitecture and Services to implement the BP, as shown in Fig. 6 (2). In this case, we generate the internal services to support the Hospital process, which are:

- Hospital: "Receive Request for Appointment" and "Register Patient data for MAS".

The first service (a) is a complex one which includes the two ServiceTasks in the Group using the patterns approach, so it has two operations; the second service (b) is generated using the basic approach, so it has only one operation. In both cases, the Software Architect has the responsibility for marking tasks that will be implemented as services, in the collaborative BP case by marking the provider as ServiceTask, and in the orchestration BP case by marking and grouping ServiceTasks, in order to generate services according to whether it is a basic or a group approach.

5. Related work

For the generation of software to support BP execution with services and software systems some proposals exist [17], mostly with partial automation. Some of the proposals can be grouped as going from BP models to UML models, such as [24], which generates UML activity diagrams (AD), use cases, collaboration and deployment diagrams from BPMN models annotated with extensions and [25], which also generate AD, use cases and analysis classes with a focus on security.

Other languages are used for services, such as [26], where these are generated based on mappings between BPMN models and SOA PI-ADL service models using process patterns. Transformation languages such as ATL [27] are also used for the generation of systems and services: in [28], to obtain Web Information Systems (WIS) from PIMs and mapping rules between models partially automating the generation, and in[29], to generate SOA models to support collaborative BPs but in WS-BPEL. It is worth noting that these languages although widely used, are not standards.

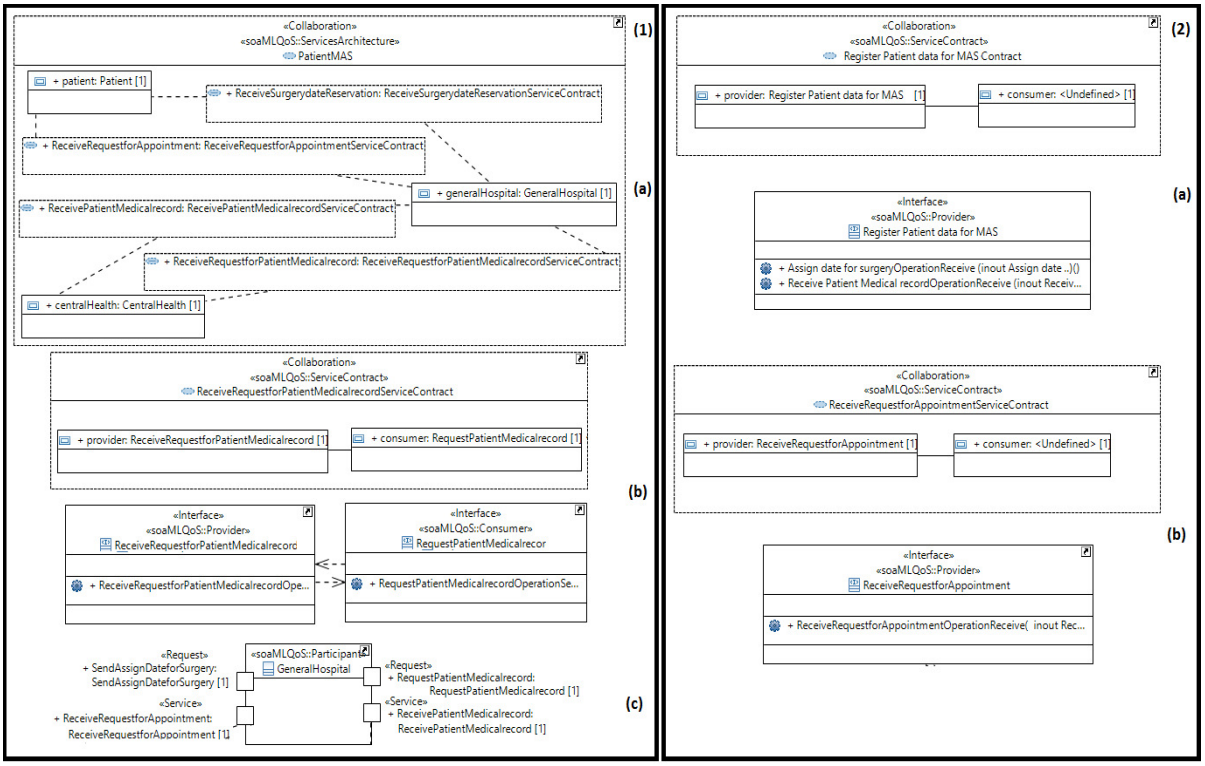

Figure 6 Generated SoaML models in SoaML Toolkit: (1) collaborative BPs, (2) orchestration BPs

Other proposals involved the definition of architectural approaches, such as in [30] where service brokers and a CIM description of the business are related, or in [31], where transformation patterns are used to design a service layer based on architectural pattern-based process integration, or in [32], where services to support BP collaborations are based on rules. Other proposals are contemporary to ours; such is the case with [33] (see [34]), which also proposes automating the navigation from BPMN2 to SoaML models, with mappings between elements, not using the QVT standard, and with limited tool support.

A key aspect of our model-driven approach which makes it different from the above is that it is based completely on standards, so making it easier to adopt in many different contexts. Another key difference is that we generate services directly from BPs, so navigating from the CIM model (requirements, i.e. BPs) to the PIM model (design, i.e. services) without any intermediate artifact or model. In addition, from the SoaML services model, we generate code, in this way providing complete traceability from the BP model to the code. Furthermore, we provide an easy-to-use tool support in the Eclipse environment.

6. Conclusions

The reference model-driven architecture we have presented aims to guide software development efforts towards implementing BPs with services in an automated way. Each model is classified within a level of abstraction in an MDA vision that helps provide traceability between business and software elements, assisting in the location of future changes and weighing their impact. We have defined mappings and transformations between elements from BPMN 2.0 and SoaML, for collaborative and orchestration BPs, to generate SOAs to support BPs Architectures, based on standards and providing adequate tool support.

We believe that environments supporting a model-driven vision on service development promote a shift of the development focus from code to models, reusing business and design patterns and hence reusing knowledge, therefore minimizing errors in service definition and implementation. This also helps improve team productivity, minimizing release time. The application of our model-driven approach based on metamodels, models and transformations between them makes it possible to lessen the effort organizations put in to align their BPs with service support, minimizing the impact of changes. The automated generation of services from BP models can be integrated into any software development methodology, such as [35].

The reference model-driven architecture is part of a cohesive framework for the continuous improvement of BPs realized by the services we provide; it is based

on execution measures of both BPs and services [36], to support the evolution and alignment of the business and software areas coherently. As current and future work we will continue to extend mappings between BPMN 2.0 and SoaML models so as to broaden the spectrum of automatic generation of services; we will also add more QoS characteristics to enhance code generation, and will integrate monitoring and other business analysis capabilities to improve BPs and services.

10. References

[1] Erl, T., Service-Oriented Arch.: Analysis and Design for Services and Microservices, 2nd Ed. Prent. Hall 2016

[2] Krafzig, D., Banke, K., Slama, D., Enterprise SOA, Best Practices, P. Hall, 2005.

[3] van der Aalst, W.M.P., ter Hofstede, A., Weske, M., Business Process Management: A Survey, 1st. Int. Conf. on BP Management (BPM), LNCS, pp. 1-12, Springer, 2003.

[4] Weske, M.: Business Process Management: Concepts, Languages, Architectures, 2nd Edition, Springer, 2012.

[5] Dumas, M., La Rosa, M., Mendling, J., Reijers,H.: Fundamentals of BPM, Springer, 2013.

[6] Papazoglou, M., Traverso, P., Dustdar, S., Leymann, F. Service-Oriented Computing: State of the Art and Research Challenge, IEEE CS, v 40(11), pp.38-45, 2007.

[7] Mellor, S., Clark, A. Futagami, T., Model driven development, IEEE Software, v 20(5), pp. 14 - 18, 2003.

[8] Bézivin, J. On the unification power of models, Software&System Modeling, v(4), pp. 171–188, 2005

[9] Stahl, T., Voelter, M., Czarnecki, K., Model-Driven Software Development: Technology, Engineering, Management, John Wiley & Sons, 2006

[10] Delgado, A., Ruiz, F., García-Rodríguez de Guzmán, I., Piattini, M.: Model transformations for Business-IT alignment: from collaborative BPS to SoaML service model, 27th ACM Symposium On Applied Comp. (SAC), 2012.

[11] Delgado, A., Towards Pattern-Based Generation of Services to Support Business Process Execution, 11th IEEE Int. Conference on Services Computing (SCC), 2014

[12] Delgado, A., QoS Modeling and Automatic Generation from SoaML Service Models for Business Process Execution, 12th IEEE Int. Conf. on Services Computing (SCC), 2015

[13] Karakostas, B, Zorgios, Y. Engineering SO Systems, a model-driven approach, IGI Global, 2008.

[14] BPMN 2.0, OMG, http://www.omg.org/spec/BPMN/2.0

[15] SoaML, OMG, http://www.omg.org/spec/SoaML/1.0.1/

[16] QoS&RT, OMG, http://www.omg.org/spec/QFTP/

[17] Delgado, A., Ruiz, F., García-Rodríguez de Guzmán, I., Piattini, M.: Main Principles on the Integration of SOC and MDD Paradigms to BPs: A Systematic Review, 5th Int. Conf. Software and Data Tech. (ICSOFT'10), Rev. Papers, 2013.

[18] Delgado, A., Ruiz, F., García-Rodríguez de Guzmán, I., Piattini, M.: Towards an ontology for service oriented modeling supporting business processes, 4th IEEE Int. Conf. on Research Challenges in Inf. Science (RCIS), 2010.

[19] Hevner, A., March, S., Park, J. , Ram, S.,Design science in Information Systems research, Journal MIS Quarterly, v 28(1), pp. 75-105, 2004.

[20] MDA, OMG, http://www.omg.org/mda/specs.htm

[21] QVT, OMG, http://www.omg.org/spec/QVT/1.1/

[22] van der Aalst, W., ter Hofstede, A., Kiepuszewski, B., Barros, A., Workflow patterns. Distributed &Parallel Databases, v 14(1), pp. 5-51, 2003.

[23] Delgado, A., González, L.: Eclipse SoaML: A Tool for Engineering SO Applications, CAiSE Forum, 26th Int. Conf. on Advanced Inf. Systems Engineering (CAiSE), 2014.

[24] Liew,P., Kontogiannis,K. Tong,T., A Framework for Business Model Driven Development, 12th Int. Workshop on SW Tech. and Engineering Practice (STEP), 2004.

[25] Rodríguez,A.; Fernández-Medina, E.; Piattini, M. Towards CIM to PIM Transformation: From Secure BP Defined in BPMN to Use-Cases, 5th Int. Conference on BP Management (BPM), 2007.

[26] Oquendo F. Formal Approach for the Development of BP in terms of SOA using PI-ADL, 4th.IEEE Int. Symposium on SO System Engineering (SOSE), 2008

[27] Jouault, F., Kurtev, I., Transforming Models with ATL (ATLAS Transformation Language), In Procs of the Satellite Events at the MoDELS Conference, 2005.

[28] de Castro, V., Marcos, E., López Sanz, M., A model driven method for service composition modelling: a case study, Int. J. of Web Eng. & Tech., v2(4), pp. 335-353, 2006.

[29] Touzi J., Benaben F., Pingaud H., Lorré J.P. A model-driven approach for collaborative SOA design. Int. Journal of Production Economics, Vol.121 (1), pp. 5-20.22, 2009.

[30] Roser,S., Bauer,B., Muller,J. Model- and Architecture-Driven Development in the Context of Cross-Enterprise BP Eng., 3rd Int. Conference on Services Comp. (SCC), 2006.

[31] Zdun, U., Hentrich, C., Dustdar, S. Modeling Process-Driven and SOA Using Patterns and Pattern Primitives, ACM Transactions on the Web, v1(3), pp. 1-44, 2007.

[32] Orriens, B., Yang, J., Papazoglou, M., A Rule Driven Approach for Developing Adaptive SO Business Collaboration, 3rd. Int. Conf. Servs. Comp., (SCC), 2006.

[33] Elvesaeter, B., Panfilenko, D., Jacobi, S., Hahn, C., Aligning business and IT models in SOA using BPMN and SoaML, 1st. Int. Work. on MD Interoperability (MDI), 2010.

[34] Delgado,0 A., I. García-Rodríguez de Guzman, F. Ruiz, and M. Piattini, From BPMN business process models to SoaML service models: A transformation-driven approach, 2nd Int. Conf. on Software Tech. and Eng.(ICSTE), 2010.

[35] Delgado, A., Ruiz, F., García-Rodríguez de Guzmán, I., Piattini, M.: Business Process Service Oriented Methodology (BPSOM) with Service Generation in SoaML, 23rd Int. Conf. on Advanced Inf. Systems Eng. (CAiSE'11), 2011.

[36] Delgado, A., Weber, B., Ruiz, F., García-Rodríguez de Guzmán, I., Piattini, M.: An integrated approach based on execution measures for the continuous improvement of BPs realized by services. Information & Software Technology v 56(2): 134-162, 2014.